小型組込み機器を対象とした 分散システムの構築法とその言語処理系の試作

盛合 智紀† 並木 美太郎‡
東京農工大学工学部 †/東京農工大学大学院共生科学技術研究院 ‡

1 はじめに

近年の組み込み機器の高性能化、低価格化を背景にセンサーネットワークが注目を浴びている。しかし、個々の機器によって利用が想定される場面が異なり、ベンダーによってハードウェアアーキテクチャの選択も異なる。これは、開発者にノードとなる組み込み機器に応じた開発環境を整備する事を求め、開発者に余計な負担を強いる事となる。

また、センサーネットワークを構築する分散システムを考えた場合、ソフトウェアアーキテクチャの選択がシステムの利便性を左右するといっても過言ではない。一般的にセンサーネットワークはクライアントサーバ型の遠隔手続き呼び出しを用いて実現する事が多いのだが、ノードのアーキテクチャの違いを吸収する事が必要になる。

そこで本研究では、組み込み機器のアーキテクチャの違いを 9P プロトコル [1] を用いて吸収する分散システムの構築法の提案と、そのシステムの為の、センサーノードの様な少ない資源しか有さない環境下でも動く VM(Virtual Machine) と言語処理系の設計と試作を行った。

2 目標

本研究では分散システムの通信プロトコルに TCP/IP を利用した 9P プロトコルを用いる事で、ノードの手続きをローカル空間でのファイル入出力の様に扱う、位置透過性の高い分散システムの構築法を提案する。また、提案した分散システムを実現する為に、数 KB 程度の RAM で動作するスタックマシン型の VM と、C 言語のサブセット的な開発言語を用いた処理系を提供する事を目的とする。

本処理系では、ホストとの通信を行う為にファイル入出力を用い、割り込みの為の API を提供する。これにより、クライアントサーバ間の通信は双方向でファイル入出力を用いる単一的なモデルを提供する事が出来る。

3 分散システムの概要

本研究ではファイル入出力を用いて RPC を実現する。システム全体で共通の名前空間を用いず、個々のクライアント毎にリモートマウントする事で、名前管理サーバが性能低下のボトルネックになる事を避ける。ノードの接続に 9P プロトコルを用い、取り込まれたノードに含まれるファイルは、手続きや引数を表すファイルで、それ等のファイルに対して入出力を行う事で引数の受け渡しや手続きの実行、戻り値の取得を行い、RPC を実現する。以下にクライアントマシンからのノードの見え方を表した分散システムのイメージを示す。

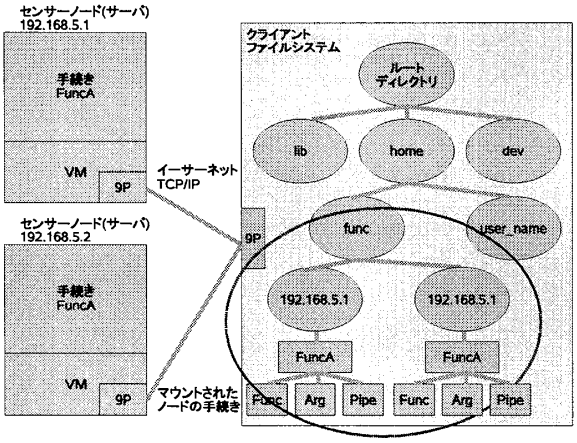

図 1: システムの構成イメージ

図 1 に示した様に、クライアントのファイルシステムを表す名前空間の部分木として、サーバの手続きが含まれる。そこで、上記のファイルに見せる為の情報としてファイル名等が別途必要になるので、本方式ではコンパイラからの情報を用いる事で解決する。サーバで動作させるコードファイルに、関数や引数の名前を埋め込む事で、サーバ上で関数名の設定等の処理を行う必要が無い事が本方式の特徴である。また、ノードの手続き FuncA を呼び出し、途中でノードへデータを送り、戻り値を取得するコードを以下に示す。

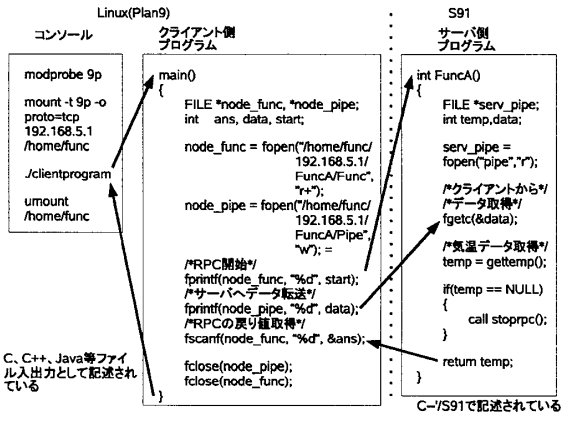

図 2: RPC の実行

図 2 に示した様に、Plan9 や Linux 等の 9P プロトコルをサポートする OS を用いて、ノードをマウントする。その後は OS のサポートにより名前管理が行われ、ファイル入出力による RPC を提供する。クライアントプログラム中の 1 回目の fprintf で RPC を開始し、最

Study on an Architecture of Programming System for Distributed Systems and Prototype of a Language Processor for the system

† Tomoki Moriai
Department of Computer, Information and Communication Sciences, Tokyo University of Agriculture and Technology
‡ Mitaro Namiki
Graduate school of Engineering, Tokyo University of Agriculture and Technology

後の fscanf で戻り値を取得する。

また、引数を与えず呼び出すだけの RPC は、cat コマンドによるファイル入出力方法によって RPC を実行する事が出来る。よって、ミドルウェアの API に沿った記述方法を用いる煩わしさや、ソケットを利用したネットワークプログラミングから開発者を解放し、位置透過性の高い分散システムを構築する事が出来る。

4 処理系の概要

本処理系では、C 言語のサブセット的な C–' という言語と、S91 というスタックマシン型の VM を元に、C–' に未実装だった制御構文とホストとの通信に必要なファイル入出力ライブラリを追加し、S91 には 9P プロトコルの通信処理機構を追加することで、RAM の使用量を抑えつつ、ホストとの連携を行う。

実行コードは実行時にモジュールとして読み込むので、ネットワーク越しに実行コードを置き換える事でノードの再利用性を高くする。また、読み込む際にコンパイラから連携して取得した手続き名を、VM 内の Plan9 ランタイムに登録する事で、ホストからの RPC に備える。

RPC の対象となる手続きは予め開発者によってランタイムに登録され、ノードに RPC の処理要求が来た場合の処理を明示的にコードに記す必要は無く、Plan9 ランタイムが割り込み処理を行う事で実現出来るのも本方式の特徴である。

5 設計

5.1 C–'

C–' と C 言語の大きな違いは以下の通りであり、この違いは処理系を簡単にする為である。

許される型は int 型 (4 バイト)、short 型 (2 バイト)、各ポインタで、配列、構造体はない。プリプロセッサの機能はなく、分割コンパイルは許さない。関数、変数は必ず定義してから使い、値が不要な関数呼出しを許さない。演算子は四則演算、関係演算子に限定され、代入は式として値を持たない。プログラム全体の終了に end 文が追加されている。

許される制御構文は、for、while、if else、continue、break、return である。また、組み込み API と RPC を指定する為の、setrpc と言うキーワードを関数宣言の関数名の前に付属する事を認める。

コンパイルすると S91 の命令セットが出力される。

5.2 S91

9P プロトコル関連の処理もしくは、割り込み処理が発生すると、S91 のメインループを抜けて割り込み処理を行い、全ての割り込み処理を消化した後に、本来の処理に戻る。また、8 バイト固定長命令の命令セットを持ち、9P プロトコルとの通信をサポートする。固定長命令にする事で、ユーザープログラムのコードサイズが犠牲になる分、処理が簡単になる為、高速な処理が行えると考えられる。サポートするオペコードは大きく 4 種類に分けられ、メモリアクセスを行う物、スタックを操作する物、制御を行う物、算術演算を行う物が存在し、全部で 36 命令である。API はスーパーバイザーコールに対応付けられる為、プラットフォームの違いを吸収する事が出来る。

5.3 組み込み API の設計

ホストとの通信を行う為に、Plan9 の様なファイル入出力のインタフェースを備えた API と、周辺機器を利用する為の割り込み処理用 API を設計する。

ノードとクライアントの連携はホストとの通信用のファイルを用いて行う。以下に API を示す。

表 1: API の設計

名前	機能
fopen(*fname,type)	通信用ファイルを開く
fclose(*fp)	通信用ファイルを閉じる
fgetc(*buf)	通信先から値を取得する
fputc(c)	通信先に値を送る
create_timer (t_handler,time,*func)	タイマーを作る
get_timer(t_handler)	タイマーの値を取得する

6 実装と評価

今回は ColdFire MCF52233 をターゲットに実装を行った。同じマイコン上の SilentC インタプリタとネイティブコードと動作性能を比較する。

また、メモリ使用量に関して、S91 の必要とする ROM サイズは 25KB 程度であり、AVR 等の 8bit マイコンでも保有し得る ROM サイズに収まっていると言える。使用 RAM サイズは S91 が 15KB 程度なので、1KB を切るには至っておらず、大きいと言わざるを得ない。

表 2: MCF52233 上での実行時間

	本処理系	SilentC	ネイティブ
3 重ループ	0.12[s]	1.19[s]	0.13[ms]
再帰呼び出し	0.03[s]	0.29[s]	0.08[ms]
和演算	0.18[ms]	1.25[ms]	0.2[μ s]

3 重ループは各要素 10 の中でインクリメントを行い、再帰呼び出しは 10 までのフィボナッチ数を求め、和演算は 1〜10 の和を計算した。全体として、本研究の方式はネイティブコードに比べ 1000 倍近い実行時間が掛かるが、インタプリタである SilentC の 10 分の 1 程度の実行時間で処理を行う事が出来る結果が出た。

7 おわりに

本研究では、Plan9 の分散ファイルシステムのプロトコルである 9P を利用した、分散システムの構築法を提案すると共に、その処理系である、C–' と S91 の試作を行った。

今後の課題としては、9P プロトコルによる RPC を実現する為の機構を実装する事や、通信や割り込みを実現する API を実装する事と、使用する RAM の量を削減する事である。

参考文献

[1] Bell-labs: Plan 9 File Protocol, 9P http://plan9.bell-labs.com/sys/man/5/INDEX.html (2008)

[2] 幸島 明男: 無線センサネットワークの研究動向 センサネットワークのためのミドルウェア技術を中心として http://www.consorts.org/šashima/paper/wsn-sashima.pdf (2004)

[3] Philip Levis, David Culler: Mate: a tiny virtual machine for sensor networks ACM SIGOPS Operating Systems Review Volume 36 pp.85-95 (2002)